**ICT-2009.3.2**-248603-IP

**Modelling, Control and Management of Thermal Effects in Circuits of the Future**

| | WP no. | Deliverable no. | Lead participant |
|---|---|---|---|
| | **WP3** | **D3.2.2** | **IMC** |

# Comparative analysis of advanced and standard CMOS circuits

| | |
|---|---|
| Prepared by | **Andrea Bartolini (POLITO)** |
| Issued by | **THERMINATOR Project Office** |
| Document Number | **THERMINATOR/D3.2.2/v1.0** |
| Dissemination Level | *Confidential* |
| Date | **05/11/2012** |

© Copyright 2010-2013 STMicroelectronics, Intel Mobile Communications, NXP Semiconductors, ChipVision Design Systems , GRADIENT DESIGN AUTOMATION , MUNEDA, SYNOPSYS , BUDAPESTI MUSZAKI ES GAZDASAGTUDOMANYI EGYETEM , CSEM, FRAUNHOFER , IMEC, CEA-LETI, OFFIS, Politecnico di Torino, ALMA MATER STUDIORUM -Universita' DI Bologna

This document and the information contained herein may not be copied, used or disclosed in whole or in part outside of the consortium except with prior written permission of the partners listed above.

## Document

| | |
|---|---|
| Title | Comparative analysis of advanced and standard CMOS circuits |
| Type | Deliverable CO |
| Ref | D3.2.2 |
| Target version | V1 |
| Current issue | V1 |
| Status | Final |
| File | |
| Author(s) | Andrea Bartolini (POLITO) |
| | |
| Reviewer(s) | Michele Carrano (ST), Alexander Burenkov (FHG) |
| | |
| Approver(s) | Giuliana Gangemi (ST) |
| Approval date | 30-10-2012 |
| Release date | 05-11-2012 |

## Distribution of the release

| | |
|---|---|
| Dissemination level | CO |
| Distribution list | |

## History

| Rev. | DATE | Comment |
|---|---|---|
| 0.1 | 27-07-2012 | Initial version |
| 0.2 | 05-10-2012 | Version for project-internal review |
| 1.0 | 12-10-2012 | Revised version, approved by all partners |
| 1.1 | 13-10-2012 | Check and ship out |

## Table of Contents

## Table of Figures

# 1. Objectives

In this deliverable (D3.2.2) a comparative analysis has been performed on different technologyl libraries to prove the effect of technology scaling on the temperature sensitivity of digital CMOS devices.

To carry out this analysis we depicted the thermal design flow presented in deliverable D3.2.1, that augment the classical physical design flow, with an intermediate step in which it is possible to apply a specific temperature to each design cells, allowing to emulate the effect of a generic thermal map in the post-place and route design. Moreover the same flow allows computing the thermal map of the target digital design induced by the device power consumption (self-heating) and evaluating the figure of merit also under this circumstance.

The figure of merit evaluated, are the critical path delay and its position, the dynamic and leakage power. This has been tested at different Vdd and Vth level showing the impact of temperature under these circumstances. POLITO has designed the comparative analysis framework and conducted the tests, whereas IMC has provided the technological library.

As we will see from the analysis the digital design is significantly affected by the temperature in all the figures of merit, and as the technology scale down its sensitivity increases, showing inverse temperature effects. As consequence of that standard design flow needs to account the temperature and its non-homogenous spatial and temporal distribution since early stage of the design.

In Chapter 2 and 3 we describe the test case used and the thermal flow properties recalling it from Deliverable 3.2.1. In Chapter 4 we applied to the design different technology and corner cases, different constant thermal map. This highlights the impact of temperature on the selected figures of merit. In Chapter 5 we go one step further and we applied to the same design different artificial thermal maps to highlight how the spatial distribution of the temperature map impacts the different figure of merit. Finally in Chapter 6, we evaluate the same design considering the thermal map self-induced by the design power consumption; this evaluates the errors that would have been introduced by a thermal agnostic design in evaluating the target figure of merit. In Chapter 7 we conclude providing some important remarks on the conducted analysis.

## 2. Test Case Description

In order to evaluate the toolset developed for thermal-variation/self-heating aware calculation of delay and power in digital circuits, we need to select and use a suitable example design.

The chosen design module should have the following characteristics:

1- It should be representative of a general logic module

2- The module's functionality and operation should be widely known by designers.

3- The module should contain balanced number of different combinational and sequential standard cells. (it should not only contain combinational gate, and not only flip flops, but a mixture of both)

4- The module should consume a reasonable silicon area so that assumption of different thermal gradient and thermal maps across chip area is reasonable.

5- The module should be publicly available. Distribution and usage of the module should not need any special license, so that we can deliver the module along with other project data without any problem.

Based on the above requirements, we have selected as the sample test case the following digital IP block:

Spiral FFT IP core [1] is a parametric customizable implementation of Discrete Fourier Transform in synthesizable RTL.

The generated FFT core performs transform on groups of 256 input samples. In each cycle 4 complex words are entered to the module. Each complex word contains a real part and an imaginary part. The width of each part is 16 bits. Calculations are done using fixed point arithmetic. Each 64 cycles a new transformation will begin. The module acts as a pipeline and has a latency of 229 cycles. The calculated outputs are again represented as 4 complex words.

**Table 1:** Selected parameter values for Spiral FFT core

| Parameter | Value |
|---|---|
| Transform Size | 256 |
| Data Type | 16Bits Fixed Point |
| Architecture | Streaming Pipe-Line |
| FFT Algorithm | Radix-4 |
| Input data width | 4 complex pairs per clock cycle |
| Output data width | 4 complex pairs per clock cycle |
| Latency | 229 clock cycles |

The above module is synthesized using Synopsys Design Compiler. The synthesized design is then entered Synopsys IC Compiler where placement, clock-tree synthesis and routing are done. The obtained design is then used in Synopsys Prime-Time for delay/power analysis. Table 1 shows the key characteristics of the synthesized Spiral FFT core. Table 2 shows the version of tool-set that we use.

In the following we give more details about each of design preparation steps. First we need to describe the standard cell libraries that we have used to create our sample chip.

Table 2 Version information for the utilized tool-set

| Tool Name | Version |
|---|---|
| Synopsys Design Compiler | D-2010.03-SP5 - Linux 64Bits - March 17, 2011 |
| Synopsys IC Compiler | D-2010.03-ICC-SP5-3, Linux 64Bits - March 11, 2011 |
| Synopsys Prime-Time | D-2010.06-SP3-6 , Linux 64Bits - August 10, 2011 |

## 2.1. Test Library Description

We have used two different technology nodes to synthesize the FFT module:

- TSMC 65nm Low Power (TCBN65LP)
- TSMC 40nm Low Power (TCBN40LP)

For each of the above libraries, 3 main categories of standard cells are available:

- Low threshold voltage cells (LVT)
- Nominal (Regular) threshold voltage cells (RVT)
- High threshold voltage cells (HVT)

For all of the standard cells, we have two library sets available which contain the data for timing/noise and power:

- None-Linear Delay Models (NLDM)
- Composite Current Source (CCS)

CCS models provide user with higher level of accuracy in estimations. Moreover, the CCS models provide the required data to enable the tool to do interpolation. In each set, libraries are characterised for very limited number of corner cases. In order to obtain delay/power in a corner case for which there is no separate characterised library the tool needs to do an interpolation. As of the Synospsy tools, this interpolation is possible using CCS libraries.

For more description of CCS please refer to [2]. Based on the above descriptions, Table 3 shows complete list of libraries that we use, along with their descriptions.

Table 3. List of library names used and their description

| Library Name | Description |
|---|---|
| tcbn40lpbwp_120b | TSMC 40nm, Low Power, Nominal VTH, version 120b |
| tcbn40lpbwphvt_120b | TSMC 40nm, Low Power, High VTH, version 120b |
| tcbn40lpbwplvt_120b | TSMC 40nm, Low Power, Low VTH, version 120b |
| tcbn65lp_200a | TSMC 65nm, Low Power, Nominal VTH, version 200a |
| tcbn65lphvt_200a | TSMC 65nm, Low Power, High VTH, version 200a |
| tcbn65lplvt_200a | TSMC 65nm, Low Power, Low VTH, version 200a |

Based on each of the libraries mentioned in Table 3 a separate chip can be created.

## 2.2. Corner Cases

Each of the available standard cell libraries are characterised for a specified set of operating conditions. The selected operating conditions usually represent one of the extremes (e.g. highest or lower temperature, highest or lower VDD) of the operation of standard cells. Table 4 lists available (more important) operating conditions for previously mentioned libraries.

**Table 4. Important characterised corners for selected technology nodes (Tech. Node 65nm LP)**

| Operating Corner | Tech. Node | Voltage (VDD) | Temperature (T) |
|---|---|---|---|
| BCCOM | 65nm | 1.32 | 0 |
|  | 40nm | 1.21 | 0 |
| LTCOM | 65nm | 1.32 | -40 |
|  | 40nm | 1.21 | -40 |
| MLCOM | 65nm | 1.32 | 125 |
|  | 40nm | 1.21 | 125 |
| NCCOM | 65nm | 1.2 | 25 |
|  | 40nm | 1.1 | 25 |
| WCCOM | 65nm | 1.08 | 125 |
|  | 40nm | 0.99 | 125 |
| WCLCOM | 65nm | 1.08 | -40 |
|  | 40nm | 0.99 | -40 |
| WCZCOM | 65nm | 1.08 | 0 |
|  | 40nm | 0.99 | 0 |

## 2.3. Design Synthesis

We use the Synopsys Multi-Mode Multi-Corner (MMCM) flow [3] for our design synthesis and implementation. So, while doing synthesis, we instruct the synthesizer to optimize the circuit for all of the defined operating corners. At each technology node, we select the most 4 important corners and use them in the MMCM flow for synthesis. For each corner case the synthesis tool is given a specific constrain file which defines timing constrains of the circuit based on that corner. Figure 1 and Figure 2 show the corners that we use during synthesis for each of our technology nodes.

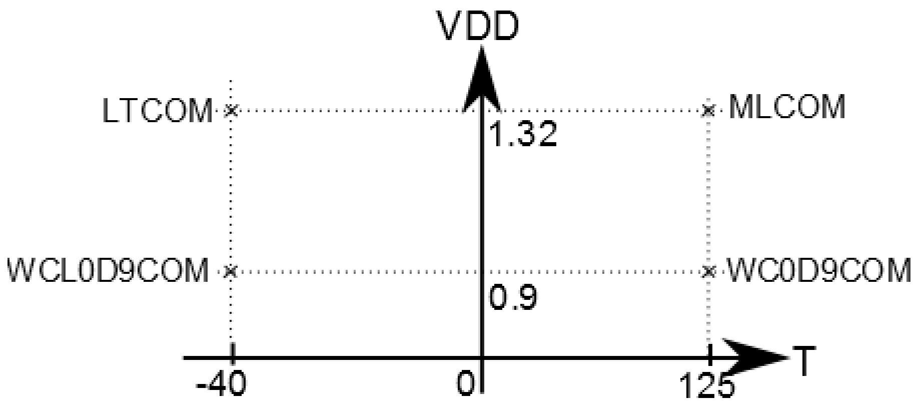


Figure 1. Used corners for Synthesis and Optimization of FFT module at 65nm

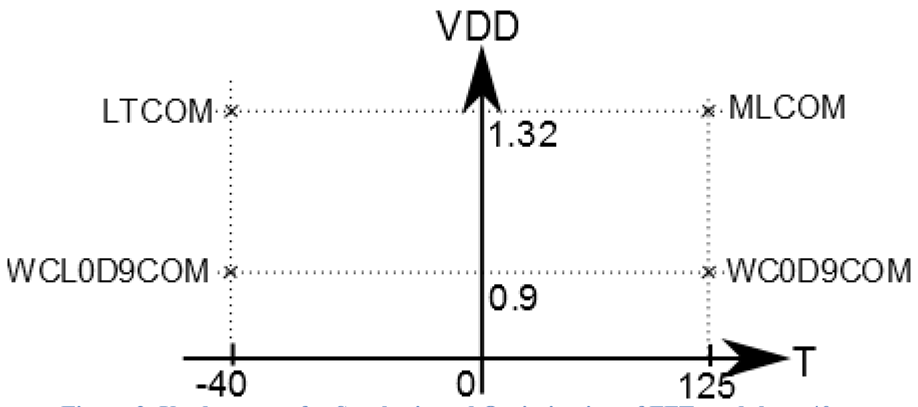


Figure 2. Used corners for Synthesis and Optimization of FFT module at 40nm

Table 5 shows basic synthesis setup for 65nm library. Please refer to [3] for more information on these parameters. For 40nm, the setup parameters are very similar to 65nm and just the file names change slightly.

Table 5. Sample std-cell library setup parameters for synthesis (at 65nm)

| parameter(s) | description | value |
|---|---|---|
| target_library | library files used during synthesis - contain std-cell information | tcbn65lpwcl0d9_ccs.db<br>tcbn65lpwcl0d90d9_ccs.db<br>tcbn65lpwc0d9_ccs.db<br>tcbn65lpwc0d90d9_ccs.db<br>tcbn65lplt_ccs.db<br>tcbn65lplt1d321d32_ccs.db<br>tcbn65lpml_ccs.db<br>tcbn65lpml1d321d32_ccs.db<br>tcbn65lptc_ccs.db<br>tcbn65lptc1d21d2_ccs.db |
| synthetic_library | library containing definitions and references to Synopsys DesignWare ready-to-use IP blocks | dw_foundation.sldb |

| | | |
|---|---|---|
| technology_file | data file used by the synthesis tool for creation of MilkyWay data-base | tsmcn65_9lmT2.tf |
| tluplus_min, tluplus_max, tluplus_map | files containing resistance and capacitance look-up table for delay calculation | cln65lp_1p09m+alrdl_rcbest_top2.tluplus cln65lp_1p09m+alrdl_rcworst_top2.tluplus star.map_9M |

Figure 3 is an example code listing for the synthesis script that we use. We have just shown the more important commands.

```
# script to be executed by:
# dc_shll-xg-t -topo

source ../tcl_65nm_MCMM_CCS/setup_dc_65nm.tcl

# create milky way database
set mw_design_library mw_fft

create_mw_lib -technology $technology_file \
-mw_reference_library $reference_lib $mw_design_library \
open_mw_lib $mw_design_library

source ../tcl_65nm_MCMM_CCS/compile_spiral.dc

elaborate fft_top_unit

create_scenario s1
set_operating_conditions WCL0D9COM -library tcbn65lpwcl0d9_ccs
set_tlu_plus_files -max_tluplus $tluplus_max \
-tech2itf_map $tluplus_map
read_sdc ../tcl_65nm_MCMM_CCS/fft_spiral.sdc

create_scenario s3
set_operating_conditions WC0D9COM -library tcbn65lpwc0d9_ccs
set_tlu_plus_files -max_tluplus $tluplus_max \
-tech2itf_map $tluplus_map
read_sdc ../tcl_65nm_MCMM_CCS/fft_spiral.sdc

# ... define other scenarios

set_active_scenarios {s1 s3 s7 s9}
report_scenarios

link

compile_ultra -no_autoungroup

## store design into a milyway database
write_milkyway -output fft_top_unit -overwrite

#... the rest of the script
```

**Figure 3. Code listing for Tcl script used during synthesis of the design**

## 2.4. Placement, Clock-tree Synthesis and Routing

Produced milky-way library by Synopsys Design Compiler is then read by Synopsys-ICC in order to continue the back-end flow. ICC is also given with the same set of scenario definitions and constraints file for each of the corners.

The following is the code-listing of the TCL script used for preparing the design inside ICC. we describe this code-listing in detail.

```
# to be run inside icc_shell
source ../tcl_65nm_MCMM_CCS/setup_dc_65nm.tcl

# required for creation of ILM
set_tlu_plus_files -max_tluplus $tluplus_max \
-min_tluplus $tluplus_min -tech2itf_map $tluplus_map

set mw_design_library mw_fft
open_mw_lib $mw_design_library
open_mw_cel fft_top_unit

current_mw_cel fft_top_unit
create_floorplan -core_utilization 0.5

place_opt
clock_opt
route_opt

save_mw_cel -as fft_top_unit_routed

create_ilm

write_def -rows_tracks_gcells -vias -regions_groups \
-components -pins -blockages -floating_metal_fill \
-nets -diode_pins -specialnets -notch_gap \
-pg_metal_fill -scanchain -output "fft_top_unit_after_routing.def"

change_names -rules verilog -hierarchy
write_verilog fft_top_unit_after_routing.v
write_sdf -version 2.1 fft_top_unit_after_routing.sdf

write_parasitics
```

**Figure 4. Code-listing of TCL script used inside Synopsys ICC**

In the shown script in Figure 4, we first load required data files and libraries by calling the setup_dc_65nm.tcl file. As described, TLUPLUS files are required for extraction of RC values in the design to be able to calculate wire delays. With set_tlu_plus_files command we set the required files.

Now we open the Milky-way library and our Milky-way design cell which is called fft_top_unit.

We then create a floorplan for the design to indicate chip boundaries for placement of std-cells. We have selected the amount of area utilization  to be 50%. This means that in the finished chip, half of the silicon area is covered by std-cells and the rest is left for performing routing and timing optimization.

We then perform placement, clock-tree synthesis and routing of the design by calling place_opt, clock_opt and route_opt commands one after another respectively.

The obtained design will be again saved as a milky-way cell. Please note that, when we saved the synthesized design into a Milky-way library in Design Compiler, we practically also saved the defined scenarios and their related timing constraints. When we open the Milky-way library inside ICC, these defined scenarios and constraints also get loaded automatically.

We are interested to be able to create larger chips by replicating one single design several times. For example, we may like to be able to create a sample chip containing 8 FFT units. For this purpose we will have 8 instances of FFT in RTL. We don't want the synthesis and place&route to happen for each of these 8 units separately. Instead we want to do this task only one time for one unit and then to be able to replicate the placed&routed design easily as many times as we like. In order to do this we create an ILM (Interface Logic Model) out of the ready design. With the ready ILM, we will treat the module like a new standard-cell. The synthesis tool and place&route tool do no more put time on synthesis, placement and routing of the contents of this new cell. They just use the available cell's contents. We use the create_ilm command to create the ILM. This ILM can be later used to create larger chips. For this report however, we focus only on one instance of FFT module.

We need to know the exact physical location of each of the cells in the design. Later we use cell locations to create the power and thermal map for our analysis. In order to save cell locations we use write_def command. With this command we store complete chip's floorplan and placement information into a DEF file. One part of this file contains design std-cell instance names, as well as their location.

Since one of our goals is to estimate power consumption of the design, we need to have switching activity statistics of different parts of the design to estimate dynamic power. In order to obtain switching activity statistics we need to be able to perform logic design simulation at gate-level. For this we need the equivalent Verilog netlist of the design as well as the SDF (Standard Delay Format) file describing all of the delay values for every element of the design. write_verilog and write_sdf commands generate these two files. These files will be then used along with the testbench to perform logic simulation at gate-level and to obtain switching activities.

For a more detailed description of the flow used in ICC please refer to [4].

## 2.5. Area/Timing reports

Table 6 shows chip area consumption for 40 and 60 technology nodes.

**Table 6. Standard-cell area at 40nm and 60nm (Nominal VTH) - (reported by Design Compiler) (Area is in um^2)**

| Technology Node | Total Leaf-cell Count | Combinational area | Non-Combinational area | Total Area |
|---|---|---|---|---|
| 65nm | 199803 | 391229 | 446817 | 838046 |
| 40nm | 137066 | 109798 | 291251 | 401049 |

**The assigned silicon area for the finished chip in the chip floorplan is shown in**

Table 7.

**Table 7. Silicon area of the finished design (obtained from the produced DEF by the Synopsys ICC)**

| Tech. Node | Dimensions (X,Y) (in um) | Die Area (in um^2) |
|---|---|---|
| 65nm | (1294.6 , 1294.2) | 1675471.32 |
| 40nm | (895.580 , 894.600) | 801185.86 |

Finally Table 8 and Table 9 show maximum running frequency of the design for each technology node for each of the (more important) corner cases. These maximum frequency values are reported by Synopsys ICC based on the routed design. ICC and Prime-Time suppose that the temperature value is uniform across the entire die area. By default, they are not capable of considering temperature gradients or variation, or any similar effect.

**Table 8. Estimated running frequency of FFT at 65nm Nominal VTH**

| Scenario Name | s1 | s3 | s7 | s9 |
|---|---|---|---|---|
| Operating Conditions (VDD, T) | WCL0D9COM (0.9, -40) | WC09DCOM (0.9, 125) | LTCOM (1.32, -40) | MLCOM (1.32, 125) |
| Minimum Clock Duration (ns) | 4.67 | 3.91 | 0.89 | 1.07 |
| Maximum Clock Frequency(MHz) | 214 | 255 | 1123 | 934 |

**Table 9. Estimated running frequency of FFT at 40nm Nominal VTH**

| Scenrio Name | s1 | s3 | s7 | s9 |
|---|---|---|---|---|
| Operating Conditions (VDD, T) | WCL0D81COM (0.81, -40) | WC0D81COM (0.81, 125) | LTCOM (1.21, -40) | MLCOM (1.21, 125) |
| Minimum Clock Duration (ns) | 12.87 | 6.31 | 1 | 1.06 |
| Maximum Clock Frequency(MHz) | 77 | 158 | 1000 | 943 |

As we can see from tables 8 and 9, for both of 65nm and 40nm technology nodes, when the value of VDD is at its maximum (VDD=1.32@65nm and VDD=1.21@40nm) with the increase in temperature, the operation speed of the circuit will decrease. (the delays become larger). However, when VDD is at its minimum (VDD=0.9@65nm and VDD=0.81@40nm) with the increase in temperature the speed of the circuit will also increase. This is called temperature inversion phenomena.

Another interesting point in tables 8 and 9 is the sensitivity of the speed of the circuit to the value of VDD at each technology node. For example, at a fixed temperature value of -40 degree, at 65nm we can see that the speed of the circuit changes from 4.67 to 0.89 ns. So basically it changes with a rate of (4,67-0.89)/(1.32-0.9)=8.99ns/V. However, at 40nm and at -40 degrees, by the change of VDD from 0.81v to 1.21v, the delay changes from 12.87 to 1.0ns. The rate of change here is (12.87-1.0)/(1.21-0.81)=29.67ns/V.

As a result, we can see that the delays in our sample design (FFT) which represents a typical logic circuit are much more sensitive to the changes in voltage at 40nm compared to 65nm. In our case the circuit at 40nm is 29.67/8.99=3.30 times more sensitive to temperature compared to the design at 65nm. This clearly emphasizes the importance of considering temperature variation as a very important factor while designing near future chips.

## 2. Tests Description

We evaluate the effect of temperature on the following design parameters (figures of merit):

1. Critical timing path delay and its on-die spatial location
2. Dynamic and static power consumption of the design.

We mainly perform 2 categories of tests:

1. Tests in which we apply a pre-determined and pre-computed temperature pattern to the design.
2. Tests in which we calculate the power consumption of the design, and then we perform thermal simulation and we obtain the design's real temperature map, we use this temperature map to measure design metrics. We then extend this test by taking into account the self-heating effect of the chip. Here we calculate power/delay values iteratively and continue the loop until convergence.

Figure 5 shows the flow of our tests in more detail. (A similar figure is also described in [5])

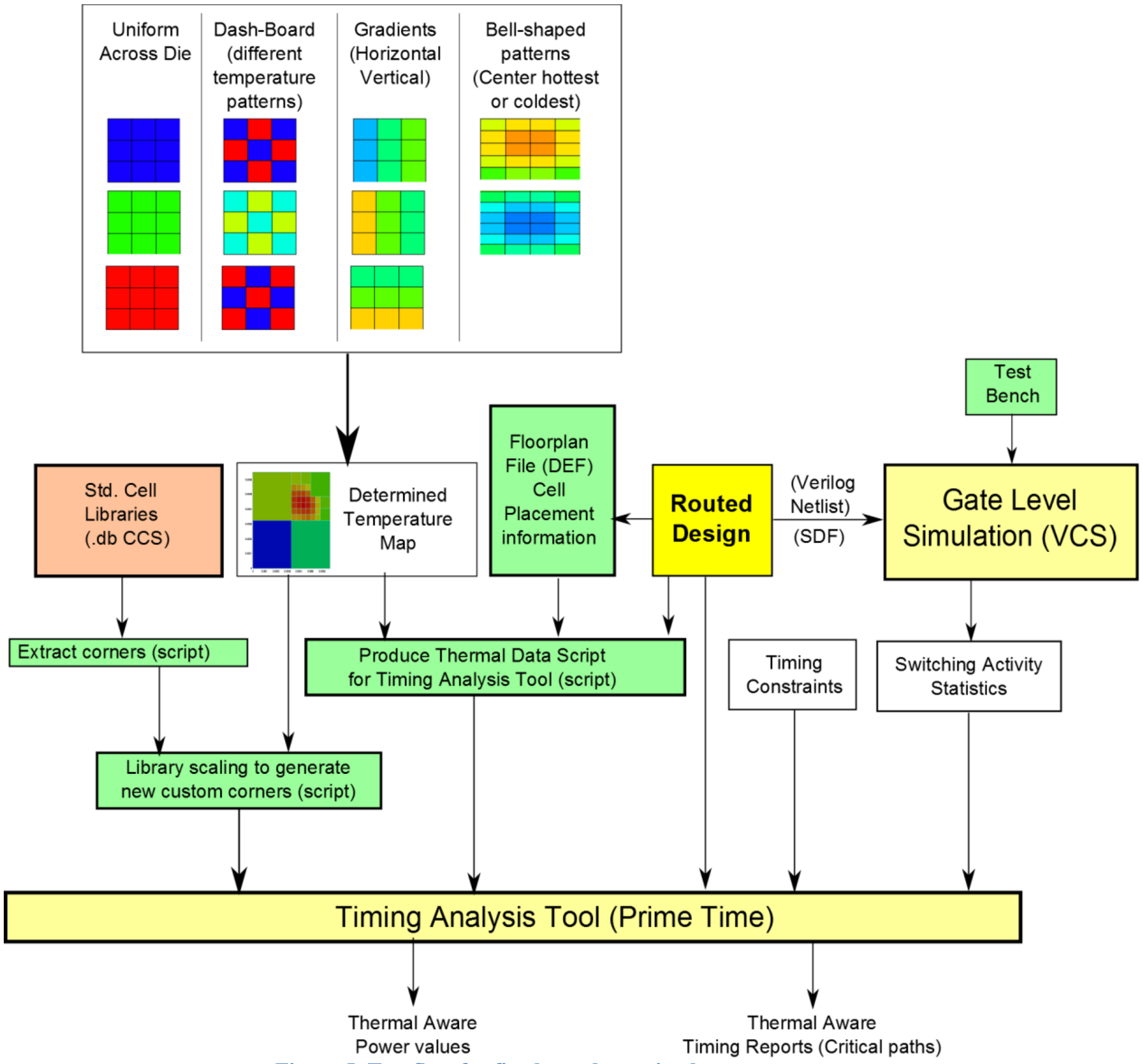

Uniform Across Die
Dash-Board (different temperature patterns)
Gradients (Horizontal Vertical)
Bell-shaped patterns (Center hottest or coldest)
Test Bench
Floorplan File (DEF) Cell Placement information
Std. Cell Libraries (.db CCS)
Determined Temperature Map
Routed Design
(Verilog Netlist)
(SDF)
Gate Level Simulation (VCS)
Extract corners (script)
Produce Thermal Data Script for Timing Analysis Tool (script)
Timing Constraints
Switching Activity Statistics
Library scaling to generate new custom corners (script)
Timing Analysis Tool (Prime Time)
Thermal Aware Power values
Thermal Aware Timing Reports (Critical paths)

Figure 5. Test flow for fixed pre-determined temperature patterns

In Figure 5, we have first defined a set of temperature patterns, comprising both corner cases and typical scenario. Determined temperature patterns that we use are 4 categories:

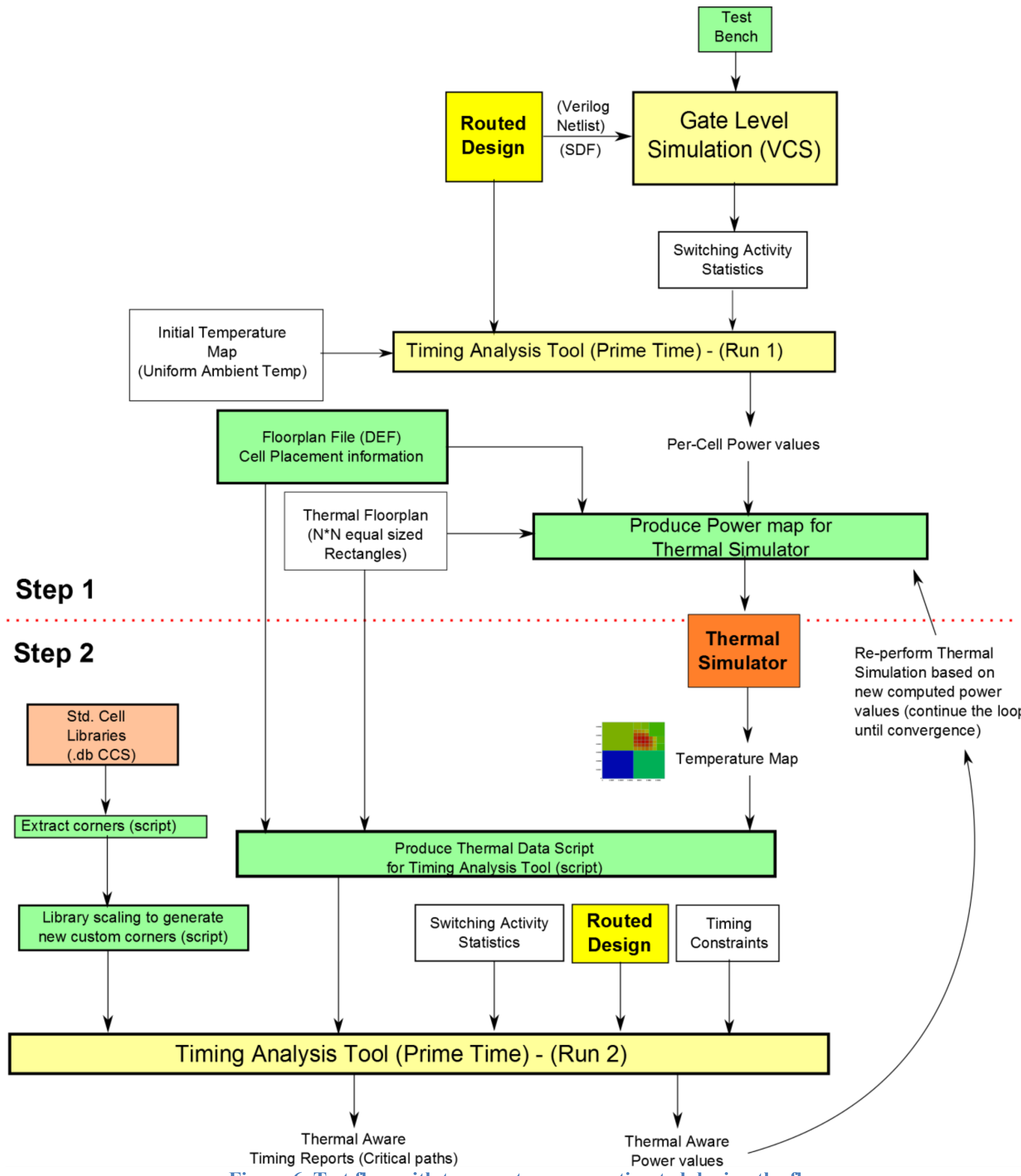
a) Uniform temperature map: The entire die area has a unique temperature value. We measure changes of figures of merit by sweeping temperature value from minimum (-40 degrees) to maximum (125 degrees). In fact, we evaluate the design for the following list of temperature values: T= {-40, -25, 0, 25, 50, 75, 100, 125}.
b) Chess-boards: Here the die area is divided into equally sized rectangles. Two temperature values are selected and assigned to rectangles to create a chess-board pattern. The temperature values are always selected so that the average temperature value of the entire die becomes a fixed value. In our tests, we have selected 50 degrees as the fixed average value. We use the following temperature value pairs during our tests: {(40, 60), (30, 70), (20, 80), (0,100)}
c) Horizontal and Vertical Gradients: Here we change the temperature value slowly from one side of the chip to another side. We use this for both of horizontal and vertical directions. We also use different temperature values and beginning and ending temperature. Again we keep the average temperature of entire die equal to a fixed value of 50 degrees. Here are the beginning/ending temperature value pairs for these patterns: T= {(65, 35), (80, 20)}
d) Bell-shaped patterns: Here we define bell-shaped patterns for temperature map of the die area. In one case, the bell reaches its maximum in the center of die area. So center is the hottest place of the chip. In another case, the bell will reach its minimum in the center and it is the coldest place. Again the temperature values are produced so that the average stays equal to 50 degree. The bell that we produce has in fact a Gaussian shape. Here are the maximum and minimum die temperature values for our different test cases: When center is hottest T={(max: 54, min: 44), (max: 63, min: 33) }. When the center is coldest T={ (max: 56, min: 46), (max: 68, min:38) }

The above patterns enter our developed toolset. Our toolset also reads the produced DEF file (described in section 1.4) to understand the location of design cells. Our toolset now produces a command script for the timing/power analysis tool, in which it indicates the operating temperature of each design cell. The timing/power analysis tool then performs its calculations based on the given temperature values.

As shown in Figure 5, the timing/power analysis tool is also given with the sample switching activity of the design. To obtain these switching activities we have performed gate-level simulation with the design test-bench, and created Verilog netlist and SDF file by the place&route tool.

The timing/power analysis tool is also given with standard-cell libraries which contain CCS models for standard-cells. (Section “2.1. Test Library Description” gave more description on CCS). This enables the tool to perform interpolation for required temperature (T) and supply voltage (VDD) values that do not exist in the library files directly.

Figure 6 shows the test flow, when the temperature pattern of the chip is not pre-determined and is computed in the flow using a thermal simulation tool.



Test Bench
Routed Design
(Verilog Netlist)
(SDF)
Gate Level Simulation (VCS)
Switching Activity Statistics
Initial Temperature Map (Uniform Ambient Temp)
Timing Analysis Tool (Prime Time) - (Run 1)
Floorplan File (DEF) Cell Placement information
Per-Cell Power values
Thermal Floorplan (N*N equal sized Rectangles)
Produce Power map for Thermal Simulator
Step 1
Step 2
Thermal Simulator
Re-perform Thermal Simulation based on new computed power values (continue the loop until convergence)
Std. Cell Libraries (.db CCS)
Temperature Map
Extract corners (script)
Produce Thermal Data Script for Timing Analysis Tool (script)
Library scaling to generate new custom corners (script)
Switching Activity Statistics
Routed Design
Timing Constraints
Timing Analysis Tool (Prime Time) - (Run 2)
Thermal Aware Timing Reports (Critical paths)
Thermal Aware Power values

Figure 6. Test flow with temperature map estimated during the flow

Figure 6 is divided into two sections: Step1 and Step2. During step 1 we calculate an initial temperature map for the design. In step 2 we use the calculated temperature map to obtain critical path delay values and power. If higher accuracy is needed we perform Step 2 iteratively, until we reach convergence for obtained values.

In step 1 the design test-bench, and the back-annotated Verilog netlist are used to obtain switching activities. Based on switching activity statistics, the timing/power estimation tool calculates power consumption of the design. In this initial step, since we do not know the actual temperature value of the die, we use ambient temperature for the operating condition of the design.

Obtained per-cell power values, along with the DEF file which contains exact design cell locations, enter our developed toolset. Our toolset produces the required power map for thermal simulation.

As of the required thermal floorplan, for doing thermal simulation, we divide the die area into equally sized rectangles. The number of rectangles is a parameter and can be selected during the test. For our tests, we always use a thermal floorplan of 4 by 4 blocks.

Reading the DEF file, for each cell, our tool determines which block in the thermal floorplan will contain this cell. We add the power value of this cell to the total power of that thermal floorplan block. We perform this process for all of the design cells. The result is the a power map of the design which can be used for thermal simulation.

For doing thermal simulation we again assume that the initial temperature of the chip is equal to ambient temperature. The output of thermal simulation is a thermal map used in Step 2.

Step 2 is basically the same as what we described in figure 4. Obtained temperature map and the DEF file containing all of cell locations will enter our toolset. Our toolset will produce a suitable script for Prime-Time to define per-cell temperature values which should be used during timing/power calculations. The timing/power analysis tool is provided with suitable library files and is enabled to do interpolation between available corners whenever required.

The temperature-map aware power values can then again return back to the beginning of Step 2 to re-calculate temperature map and to re-estimate delay/power values. This time the initial temperature of the chip is no more the ambient temperature. But it is the temperature map that we have obtained during the previous phase of simulation. Moreover, the power map is updated with new per-cell power values.

Our toolset supports the entire loop in an automated way. Meaning that it performs all of the execution and data handling required to do the iterations automatically.

Currently our toolset is using Hotspot thermal simulator to perform thermal simulation. However, there is no limitation in replacing Hotspot with other available thermal simulation packages.

## 2.1. Developed tool-set and flow

To perform the specified tests we have developed a tool-set which mainly do the following set of tasks:

1. Generates required power map for thermal simulation
2. Enables delay/power analysis tool (in our case Synopsys Prime-Time) to:
    a. Handle non-uniform temperature map for delay and power calculation
    b. Perform interpolation between characterised corner cases, when required. (this one is mainly a feature of the tool itself, and our contribution here is just to perform a correct setup of the tool and libraries)
3. Produce suitable reports to visualize the results. Most specifically, conversion of power/timing reports produced by the timing/power analysis tool into suitable plots.

Our developed tool-set has two basic characteristics:

1. It is compatible with standard ASIC design flow file formats
2. It seamlessly integrates into the ASIC design flow.

Figure 7 shows the inputs/outputs of our "thermal power map generator". As described in Figure 6, this is a part of our tool-set which produces the required power map for thermal simulation.

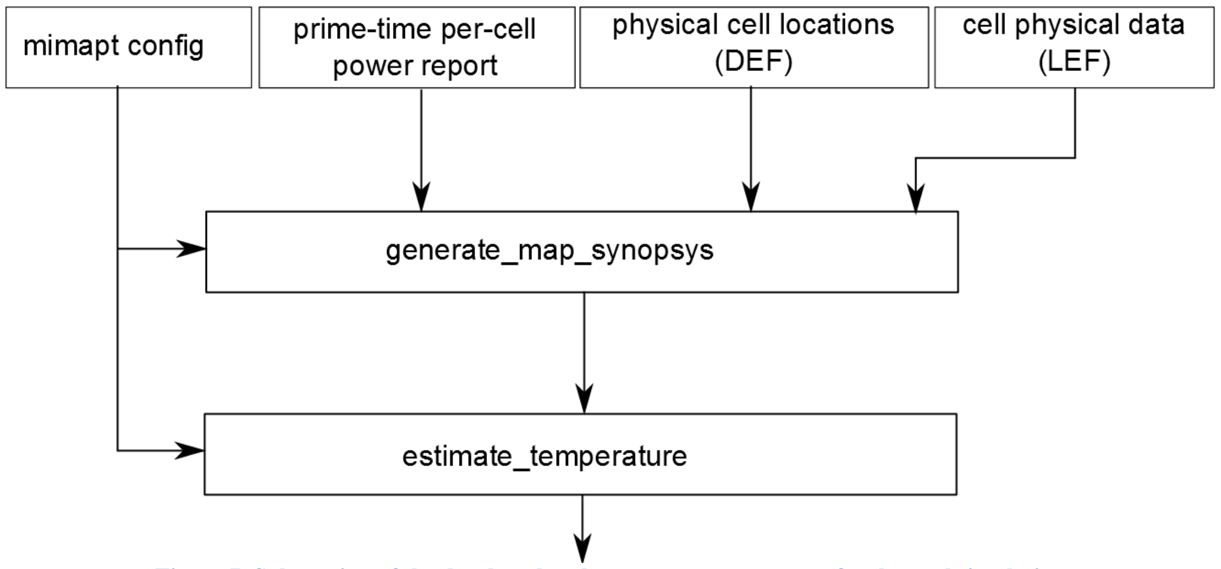


Figure 7. Sub-section of the developed tool-set to prepare contents for thermal simulation

As we can see, Figure 7 has two main blocks: generate_map_synopsys and estimate_temperature. The first block receives the per-cell power report produced by prime-time, the physical cell locations and standard-cell library physical data (in standard LEF format) and produces a data-base which for each cell contains complete cell information. (its location, size, static/dynamic/total power) The second block, uses this data-base along with the configuration data specified in mimap_config (which describes the number of blocks in the thermal floorplan) and produces all of the required material to run the thermal simulation package.

Figure 8 shows the inputs and output of the second important part of our tool-set. The block which enables the timing/power analysis tool to handle a non-uniform temperature map for timing/power estimation is shown in the figure.

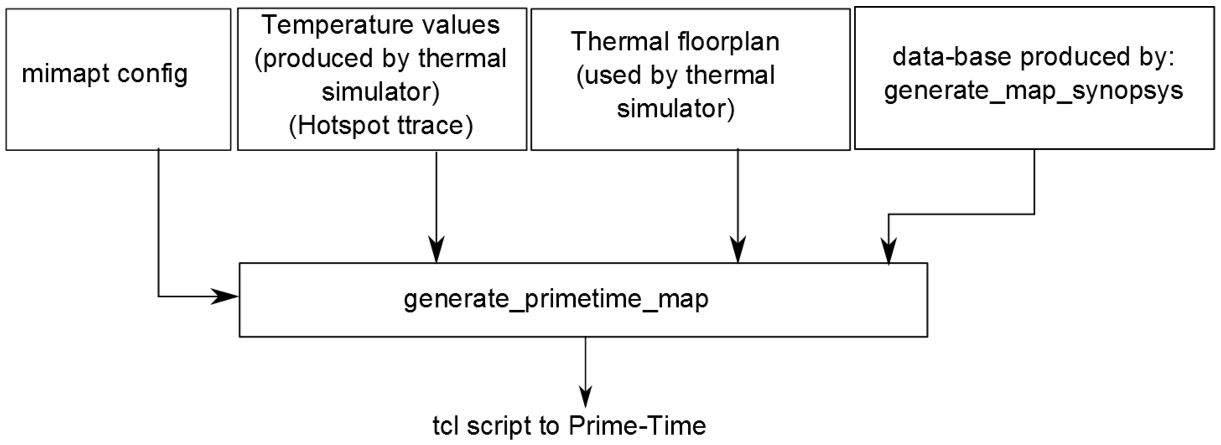

mimapt config
Temperature values (produced by thermal simulator) (Hotspot ttrace)
Thermal floorplan (used by thermal simulator)
data-base produced by: generate_map_synopsys
generate_primetime_map
tcl script to Prime-Time

Figure 8. Sub-section of the developed tool-set to define the non-uniform temperature map for timing/power analysis tool

As we can see in Figure 8, we are feeding the generate_primetime_map with two sets of basic informations: a) temperature map, which consists of two files, one that contains temperature values and the other which contains the dimensions of the floorplan blocks and their locations b) the data-base which is produced by generate_map_synopsys.

Using this data, our tool produces a TCL script for Synopsys Prime-Time, which defines for each design cell its operating temperature separately. Prime-Time will then use this operating conditions in order to calculate power and delay for that cell.

Finally, a sub-section of our tool-set is dedicated to reading the timing/power reports produced by Prime-Time and generating suitable Matlab scripts and data files to be used for futher analysis of results. Table 10 contains a list of these sub-sections, their description of operation as well as their inputs/outpus.

**Table 10. Developed routines for data visualization in Matlab**

| Routine Name | Description | Inputs | Outputs |
|---|---|---|---|
| *generate_matlab_floorplan* | Generates a data-file which can be read in Matlab and contains the temperature map | - Thermal Floorplan file (as used by Hotspot)<br>- Thermal simulation output (ttrace file format of Hotspot) | Text file, containing dimensions and temperature values to be read inside Matlab |
| *create_power_matlab* | Reads a set of power reports produced by prime-time and extracts total internal power, total switching power, leakage power and total power | - Power reports produced by prime-time | Text file, containing extracted values to be read inside Matlab |
| *create_critical_path_matlab* | Reads the timing critical path report of Prime-Time. Produces an array indicating on-die physical location of critical path | - Timing critical path report (Prime-Time)<br>- Data-base produced by *generate_map_synopsys* | Text file, containing physical location (X,Y) of cells in each critical path |
| *create_slack_matlab* | Reads the timing critical path report of Prime-Time. Extracts the slack value of each critical path. | - Timing critical path report (Prime-Time) | Text file, containing critical path delay values to be read by Matlab |

In Table 10 , the only routine which has really a complex operation is create_critical_path_matlab. This routine enables us to see the physical location of the critical path(s) of the design inside Matlab. So, we will be able to better evaluate the effect of changes in temperature on the spatial location of timing critical path. This routine receives the timing critical path report generated by Prime-Time. This text report shows the user which paths in the logic design have timing problems. For each path, it represents full description of path elements, the type of element, and its delay. The number of analyzed paths are controlled by user. For our test flow we always analyze the first 10 critical paths of the design. Our developed routine is capable of understanding the timing critical path report of Prime-Time. For each cell instance in the report, our routine looks up the data-base produced by generate_map_synopsys and extracts the spatial location of the cell. It then produces a text file containing spatial location (X, Y) of each cell in the critical path, along with its delay value.

## 3. Constant Thermal Map

We now represent the results for the test cases in which the temperature value is equal across the entire die. Our results contain the following set of information:

a) Design dynamic power, leakage power, total power. (For a detailed definition of these power components refer to [6])
b) Worst timing critical path slack value.
c) The temperature-map of the chip and the physical location of first 10 critical paths of the design.

We present the results for following cases:

a) Design is created using TSMC 65nm LP Nominal VTH Library and VDD=0.9V.
b) Design is created using TSMC 65nm LP Nominal VTH Library and VDD=1.32V.
c) Design is created using TSMC 40nm LP Nominal VTH Library and VDD=0.81V.
d) Design is created using TSMC 40nm LP Nominal VTH Library and VDD=1.21V.

For all of the power calculations, we suppose that FFT unit is running at 75MHz.

Before continuing to present results, we first clarify the units used for representing numbers in the plots. The code listing shown in Figure 9 is the output of report_units command inside Synopsys environment.

```
pt_shell> report_units
****************************************
Report : units
Design : fft_top_unit
Version: D-2010.06-SP3-6
Date   : Fri Aug 31 16:50:37 2012
****************************************


Units
---------------------------------------------
Capacitive_load_unit           : 1e-12 Farad
Current_unit                   : 0.001 Amp
Resistance_unit                : 1000 Ohm
Time_unit                      : 1e-09 Second
Voltage_unit                   : 1 Volt
Internal_power_unit            : 0.001 Watt
Leakage_power_unit             : 1e-09 Watt
```

**Figure 9. units used for representing power/delay values in Synospys Prime-Time (TSMC 40nm Library)**

Based on Figure 9, dynamic power is mainly represented in mili-Watts and delay in nano-Seconds. The major part of total power is dynamic power. And the major part of dynamic power is internal power. Static (leakage) power is very small in comparison with dynamic power. As a result, In reports, produced by the tool leakage power numbers are in pW.

Each of the figures that we show in this section has 5 rows:

1. First row is total dynamic power
2. Second row is total static power
3. Third row is total power
4. Fourth row is the slack value of timing critical path. Please note that this is the slack value. Meaning that: it is the difference between the specified maximum delay in the timing constraints file and the actual delay of longest path in the current circuit. For all of

the results that we show here (for both 40nm and 65nm, for all the cases) a single constraint file is used. The constraint file specifies a running frequency of 75MHz for the FFT module. (The specified path delay is 13.33ns) A positive slack value means that, the actual circuit is running faster than what is requested in the constraint file. The bigger positive values mean higher running speed. A negative slack value means the circuit is not capable of running with the requested frequency.

5. The final row is the temperature-map of the design, and the location of 10 first timing critical paths on it. Standard-cells creating a critical timing path, are shown with '*' symbol. Standard-cells of each critical path are connected together with straight lines.

In all of the plots, we provide a horizontal line which represents the value of the parameter shown in the plot when the entire die area is at 50C. This can be used as a kind of reference for later comparisons. Please note that, in all of the fixed temperature map patterns that we have used during all of our tests, the average value for temperature of the die is 50C.

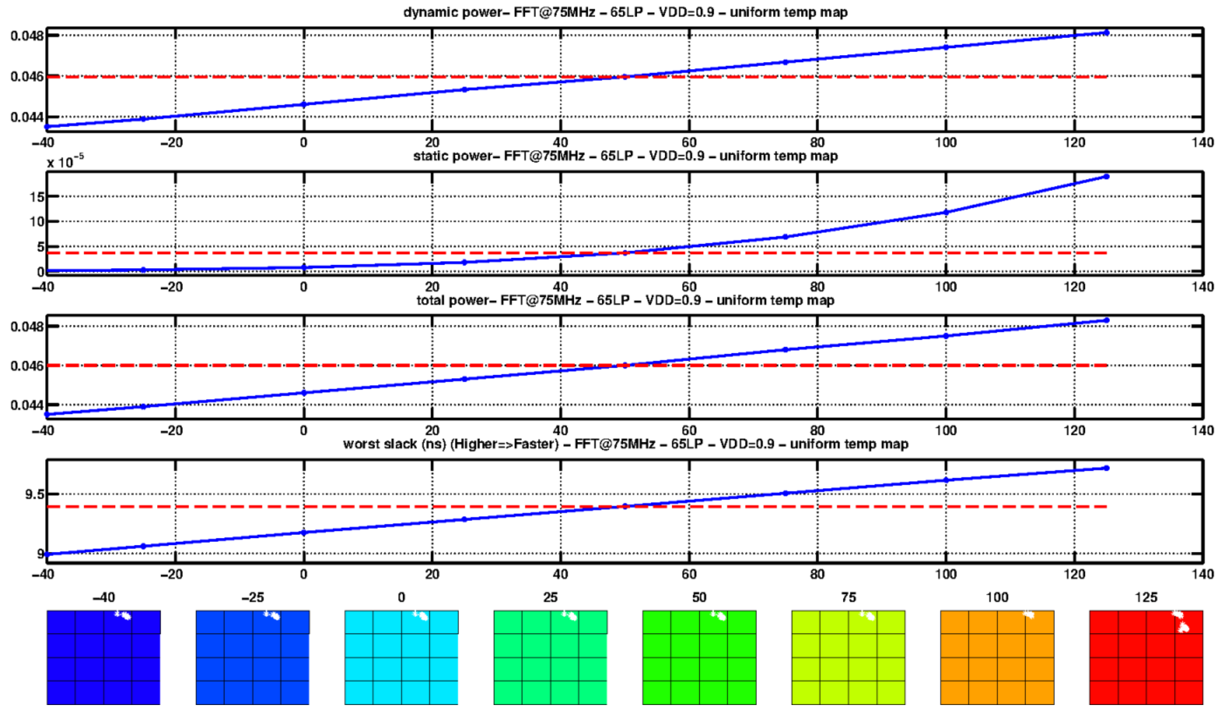


Figure 10. delay/power - TSMC 65nm Nominal VTH - unique temperature across die area - VDD=0.9v - Temperature sweep from -40 to 125 C.

As we can see in Figure 10, both of dynamic and static power values change with temperature. They both increase with the increase in temperature value. However, the percentage of change for leakage is much higher compared to dynamic power. For dynamic power the amount of change in power value relative to its initial value at -40C is 6.7%. For leakage power the amount of grow relative to initial value at -40C is 12052%. This high percentage is due to the fact that, at -40C the amount of leakage current is very small and then grows exponentially.

Figure 10 shows that the slack is improving with the increase in temperature value. This means that the circuit can run at higher clock frequencies when we increase the temperature.

Finally we can look at the physical on-die locations of first 10 timing critical paths for each temperature value. The physical location does not changed significantly between different temperature values however; it is not also the same for all of them. This shows that with the increase in temperature, the amount of delay in different standard-cells, changes with different rate. As a result, a path which has not been critical at -40°C will become a critical path 125°C.

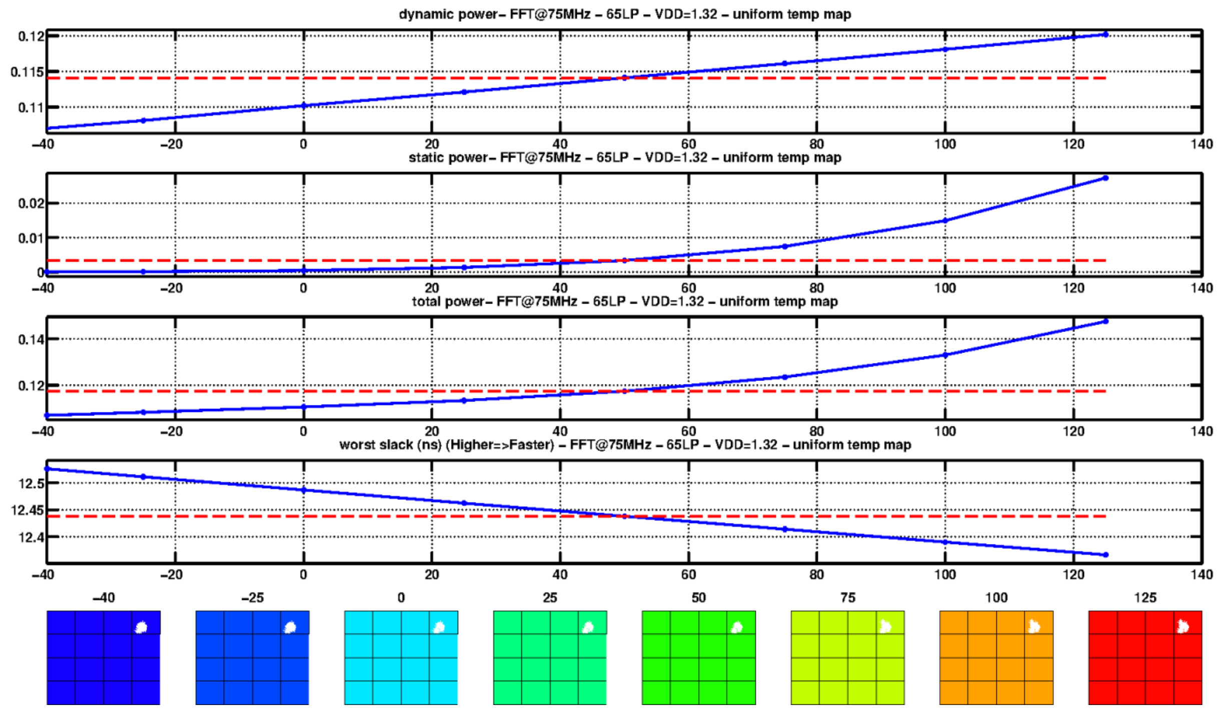


**Figure 11. delay/power - TSMC 65nm Nominal VTH - unique temperature across die area - VDD=1.32v - Temperature sweep from -40 to 125 C.**

Figure 11 shows design parameters when we increase the supply voltage from 0.9v to 1.32v. We can see similar trends with what we saw in Figure 10 for power values. However, as can be seen the total power plot in Figure 10 is almost linear, however in Figure 11 total power curve shows as an exponential. This is because at VDD=0.9v, the amount of leakage power was very small and dynamic power was completely dominant. However, with the increase in VDD to 1.32, the effect of leakage gains more and more importance.

The important point on Figure 11 is the trend of slack value. As we can see, in contrary with Figure 10, when we increase the temperature, slack will decrease. Meaning that, by the increase of temperature, the circuit can run in lower frequency values.

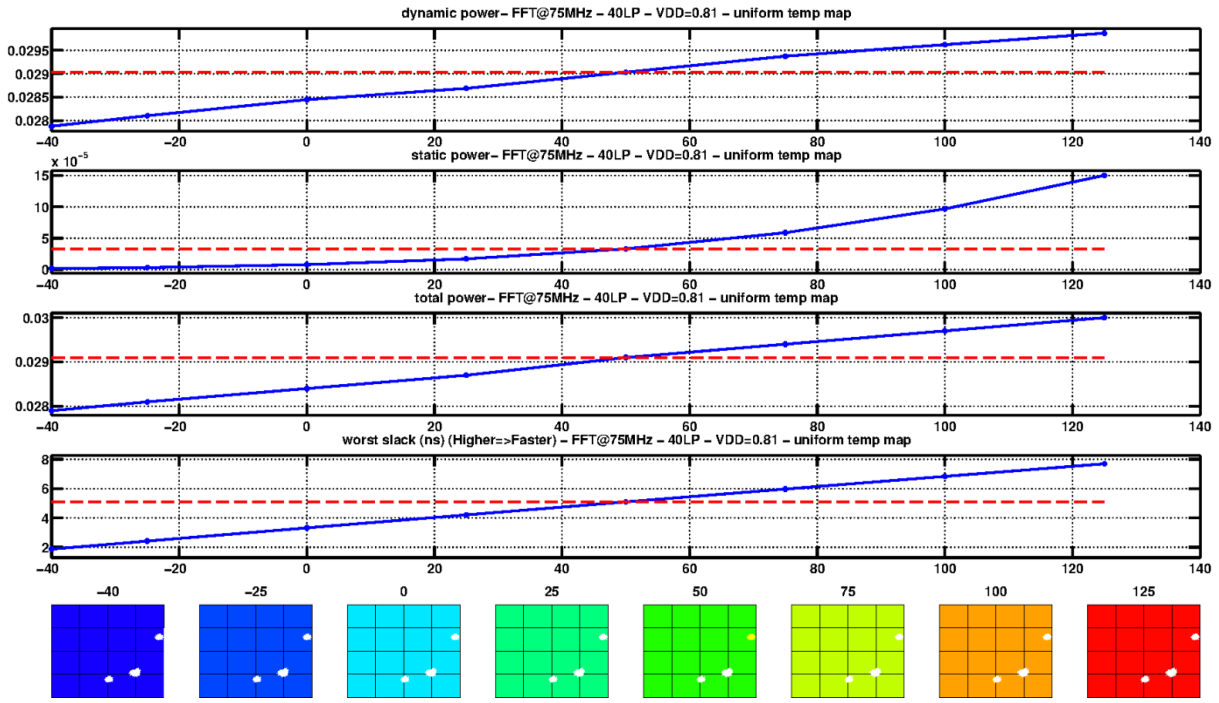


Figure 12. delay/power - TSMC 40nm Nominal VTH - unique temperature across die area – VDD=0.81v - Temperature sweep from -40 to 125 C.

The same as Figure 10, we can see in Figure 12 that by the increase in temperature the slack will improve. Again by the increase in VDD to the maximum, we should expect the opposite trend. As shown in Figure 13.

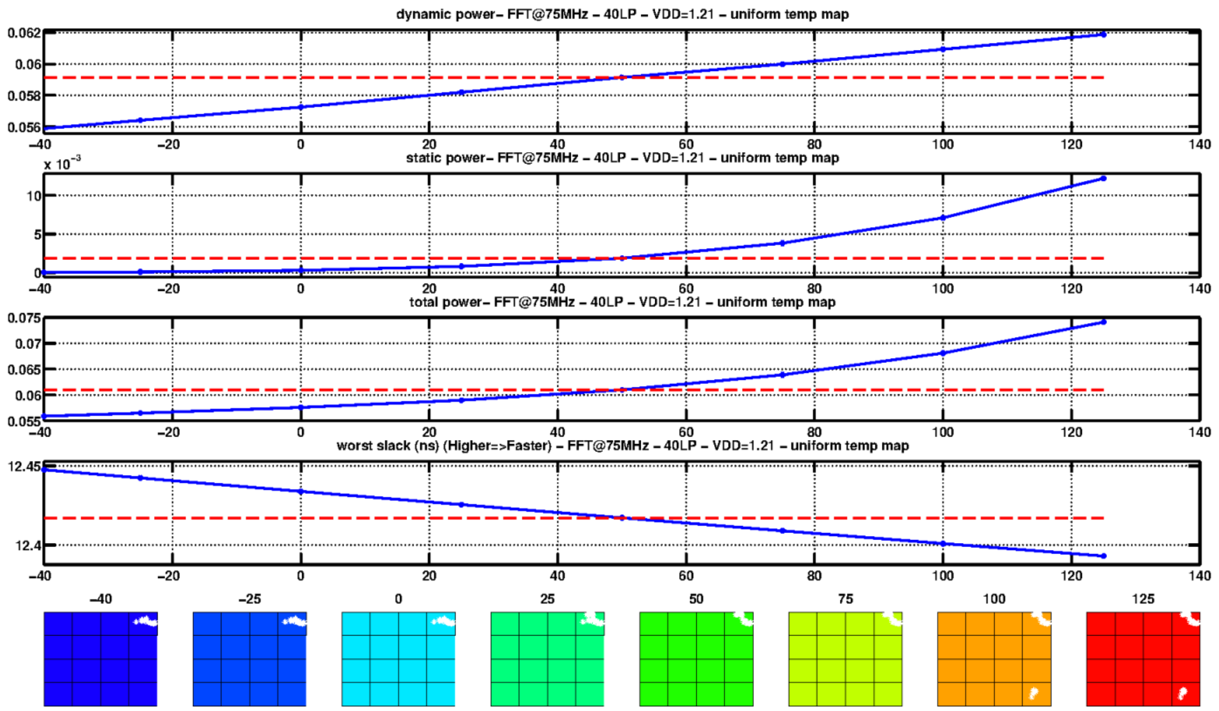


Figure 13. delay/power - TSMC 40nm Nominal VTH - unique temperature across die area – VDD=1.21v - Temperature sweep from -40 to 125 C.

Finally, we are going to show the changes in power and slack value for different supply voltages in the temperature range of -40C to 125. We show the results for 65nm and 40nm designs in one plot so that they can be compared.

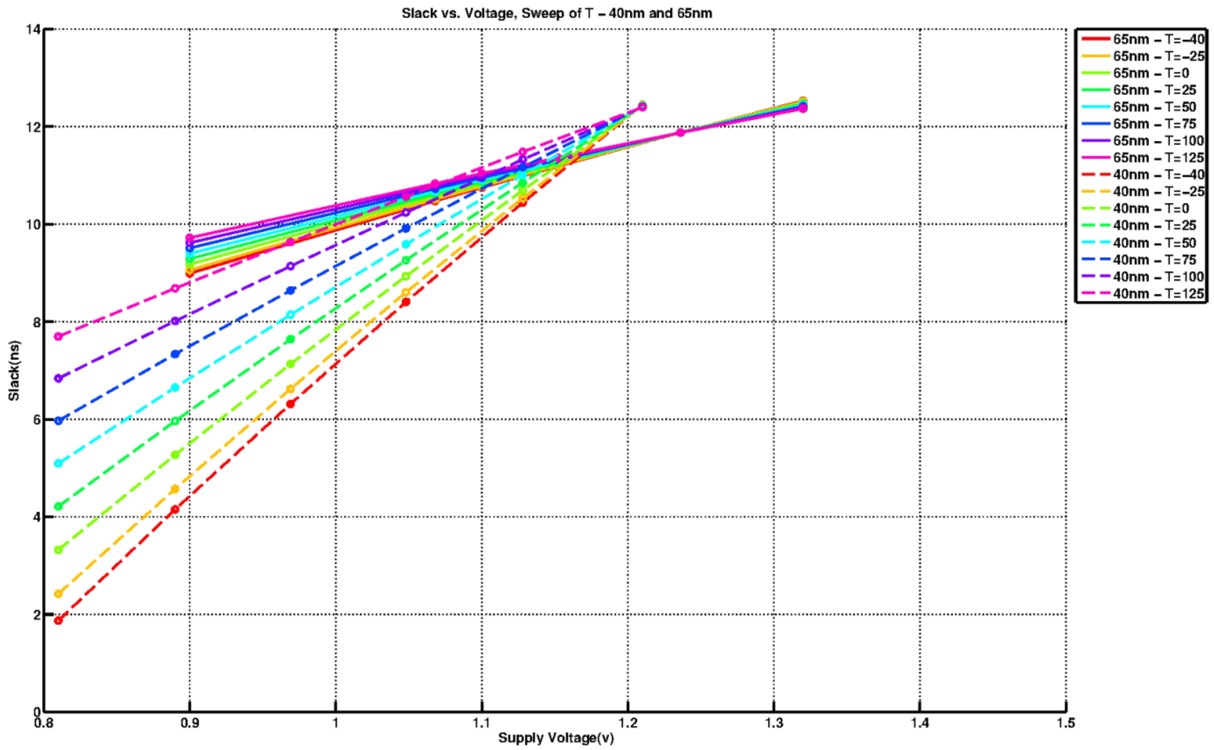


**Figure 14. Changes in slack value of the FFT circuit with the change in VDD and temperature, 65nm (solid lines) and 40nm (dashed lines)**

In Figure 14, X axis is VDD and Y axis shows the values of slack. The plot is showing slack values for both of 65nm and 40nm technology nodes. For each temperature value, the plot shows changes in slack by the change in VDD. As we mentioned before the range of valid values for VDD at 65nm is between 0.81v and 1.21v, however for 65nm VDD changes between 0.9v and 1.32v.

As we can see in Figure 14, for both of 65nm and 40nm technology nodes, there exists a voltage, in which for all of the temperature values the slack of the circuit is almost the same. This means that for both technology nodes, there exist a supply voltage value in which the slack remains almost unique during the entire range of temperature. In order to show this point clearly, we zoom into one of the areas in which all of the slack value lines cross each other. Figure 15 shows this point for 65nm.

As we can see in Figure 15, in the area between VDD=1.232 and VDD=1.242, all of the slack values lines cross each other. Please note that the cross point is not a unique signal point. But it happens in a small range of voltage values. For example, as we can see, for the slack values line at -40C, before VDD=1.232v, this line has the lowest values and after VDD=1.242v, it has the highest values compared to other lines.

Looking at Figure 14 again, we can have a comparison between the speed of FFT unit at each of 65nm and 40nm technology nodes. As we can see, for a range of VDD values, the circuit at 65nm is faster than the circuit at 40nm. At T=125C, as we can see the 40nm design is slower than 65nm design until VDD reaches approximately 1.05v. At T=-40C, this point is around

1.15v. After VDD=1.15v, the 40nm design is faster than 65nm design for every temperature value. Figure 16 is a zoomed-in version of Figure 14. In Figure 16 we have zoomed on the area in which slack lines of 40nm design cross slack lines of 65nm design.

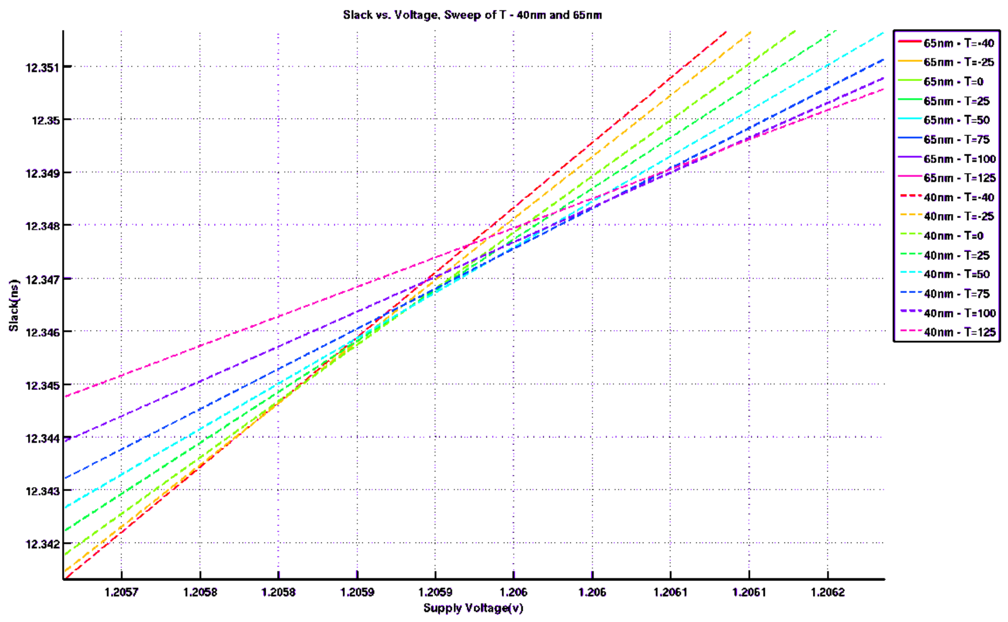


Figure 15. Zoomed-in version of Figure 14. Slack values vs. VDD for different T, zoomed on cross-point of all slack lines for 65nm.

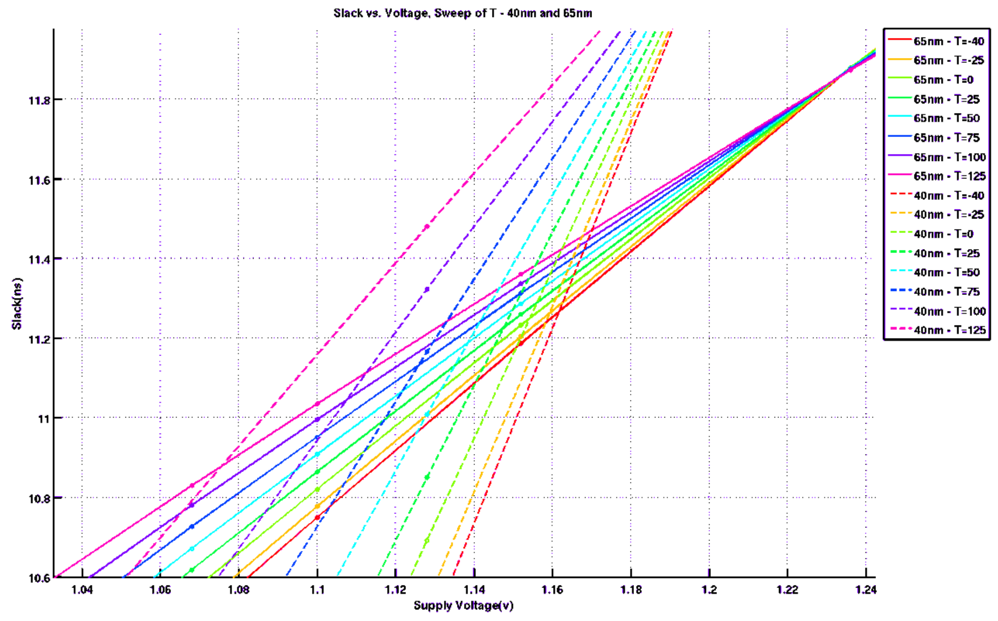


Figure 16. Zoomed in version of Figure 14. slack values vs. VDD for different T, zoomed on cross-point of all slack lines for 65nm (solid lines) and slack lines for 40nm (dashed lines).

We now represent comparisons between consumed power values of the FFT circuit at each of 65nm and 40nm technology nodes.

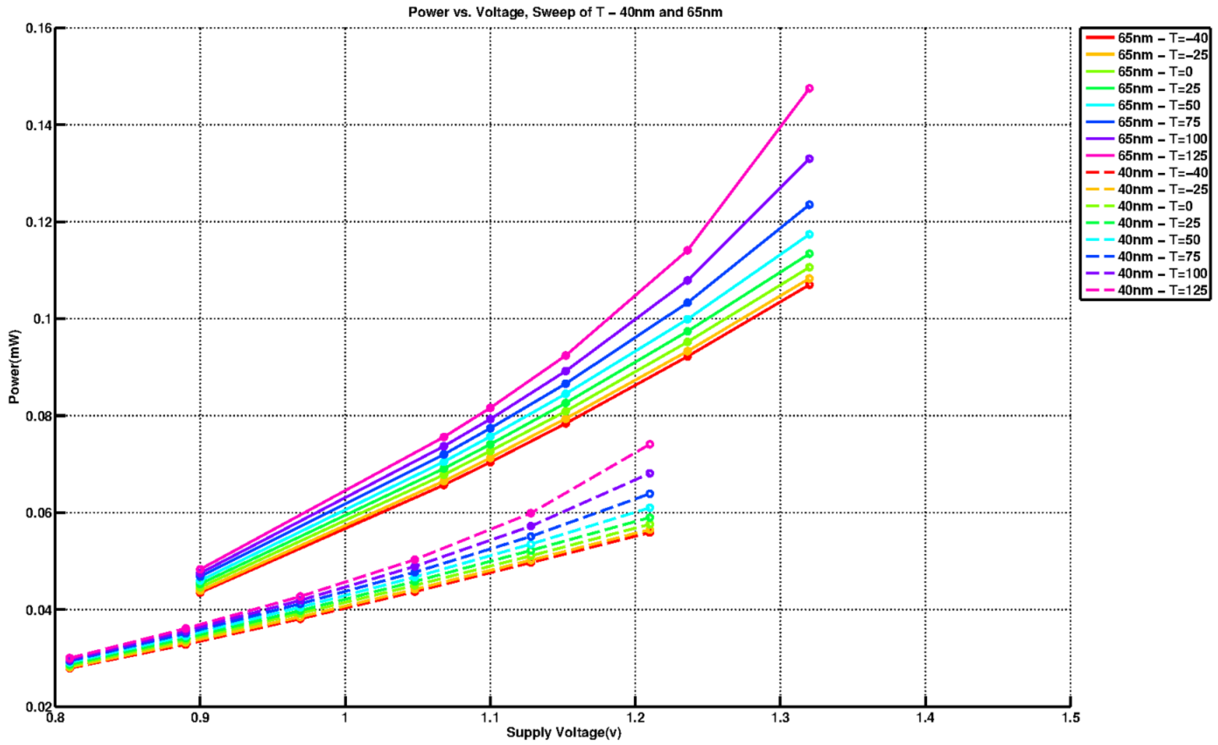


**Figure 17. Consumed total power, FFT @ 75MHz, for 65nm and 40nm nodes**

In Figure 17, as we can see, X axis represents the supply voltage and Y axis represents power. Each curve shows the changes in power with the change in VDD for a specific temperature value. As we can see, in terms of power consumption, the 40nm design is superior to 65nm in all of the situations.

Now we evaluate the amount of energy (product of power and cycle time) which is consumed by the design at each of the test points. For each test point (VDD and T pair) for each of 40nm and 65nm technology nodes, we have the maximum running frequency of the circuit, and also we have power consumption of the circuit at 75MHz. Since dynamic power scales linearly with running frequency of a circuit, using power element values at 75MHz we first calculate the total power of the circuit at maximum running frequency for each point. We now calculate energy for each test point by multiplying calculated power and the clock cycle time of the test point. Figure 18 shows the calculated per-clock-cycle energy of the FFT design for 40nm and 65nm for different test points.

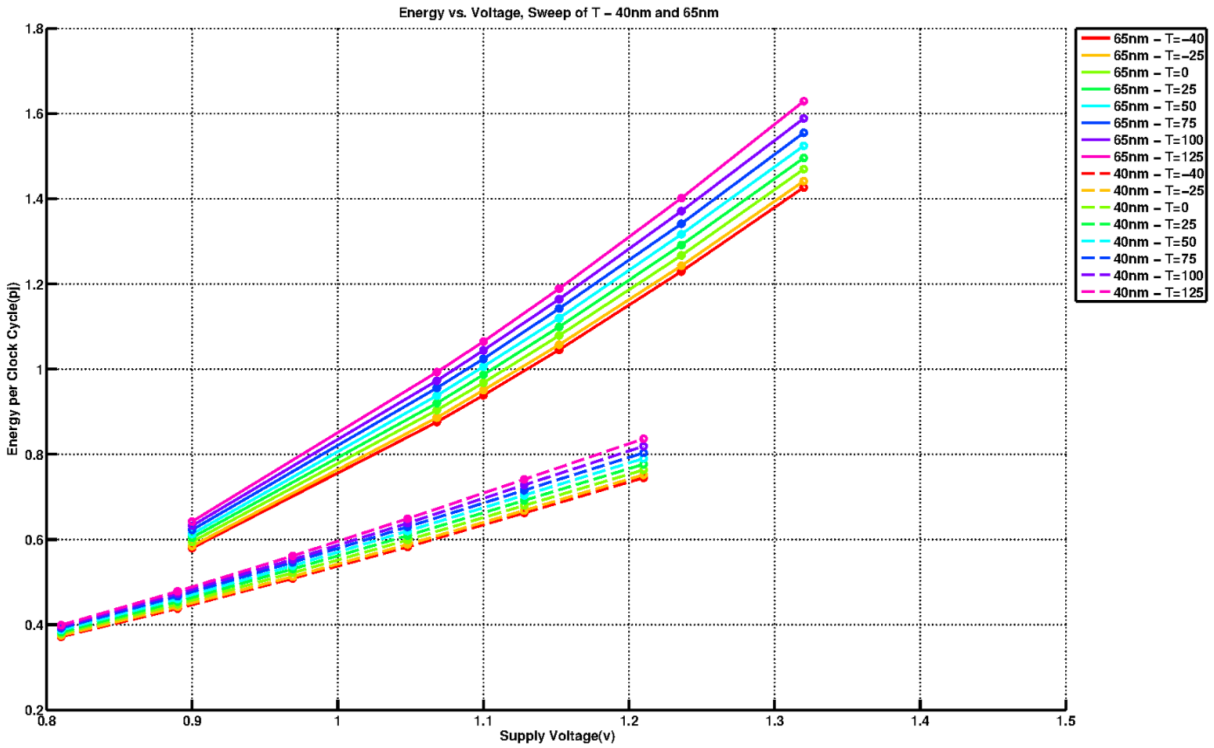


**Figure 18. Per clock-cycle energy of the FFT design for the entire test points for 65nm and 40nm.**

## 4. Variable Thermal Map

We now represent the results when temperature across die area is not unique. As described in section 2, we will evaluate these cases: a) Chess-board patterns. b) Horizontal and Vertical Gradients and c) Bell-shaped patterns.

The following sample figures show changes in design metrics with different temperature maps.

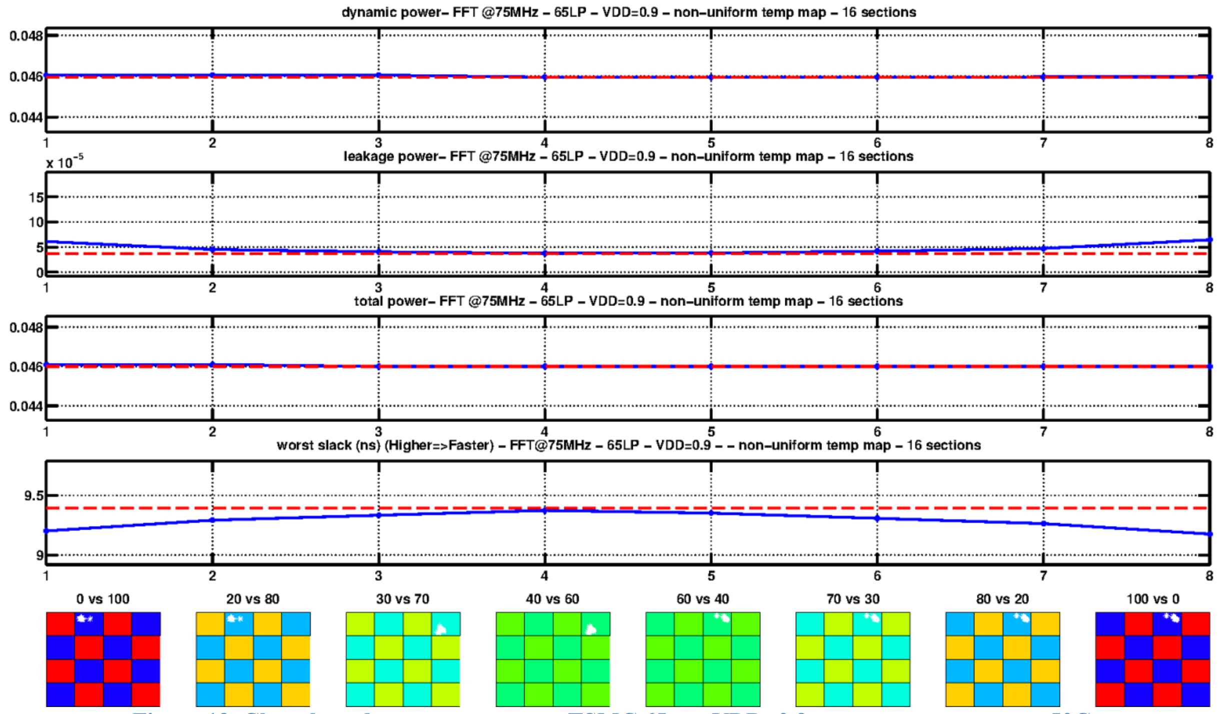


Figure 19. Chess-board temperature map, TSMC 65nm, VDD=0.9v, average temperature 50C.

In Figure 19 we have applied a set of chess-board temperature patterns to the design. Different chess-board temperature patterns have difference temperature gradient values. The amount of temperature gradient is shown at the top of each floorplan in the final row of the figure.

As we can see, the amount of dynamic power for different patterns is almost the same as the reference (50C). This is because, as we noted before, dynamic power changes linearly with temperature. However, for leakage power, as we can see, when the amount of temperature gradient is increased, the leakage power value gains clear difference compared to leakage at 50 degrees.

Based on Figure 10, leakage power value at 50C is equal to 3.695e-5. For the chess-board with averaged temperature of 50 degrees and temperature gradient of 100 degrees shown in Figure 19, the amount of leakage grows to 6.456e-5. This means that leakage has a difference of more than 74% relative to its reference value.

This tells us, if we are calculating leakage for a sample design, and if we use the unique averaged temperature value for entire die area (as what traditional methods do), the estimated value for leakage can have very large errors (as we noticed more than 50%).

Looking at the slack value, we can see that, again although the average temperature is 50 for all of the cases, the slack value of the design cannot be calculated just based on average temperature. The slack value represented in Figure 10 for uniform 50 degree chip is 9.396ns, however looking at Figure 19, the slack value for the chip with average temperature of 50 but gradient of 100 degrees is around 9.17ns.

This tells us, for correct estimation of slack value; it is required to do power/timing analysis considering on-die temperature variation.

Looking at the physical location of timing critical paths, we can see that they usually happen in the cold areas of the chip. This is in line with our previous observations that for VDD=0.9v, the design speed will be worst at lowest temperatures and will improve by temperature increase. Figure 20 shows the same chess-board at VDD=1.32v. As we can see, timing critical paths happen in hotter areas of the chip.

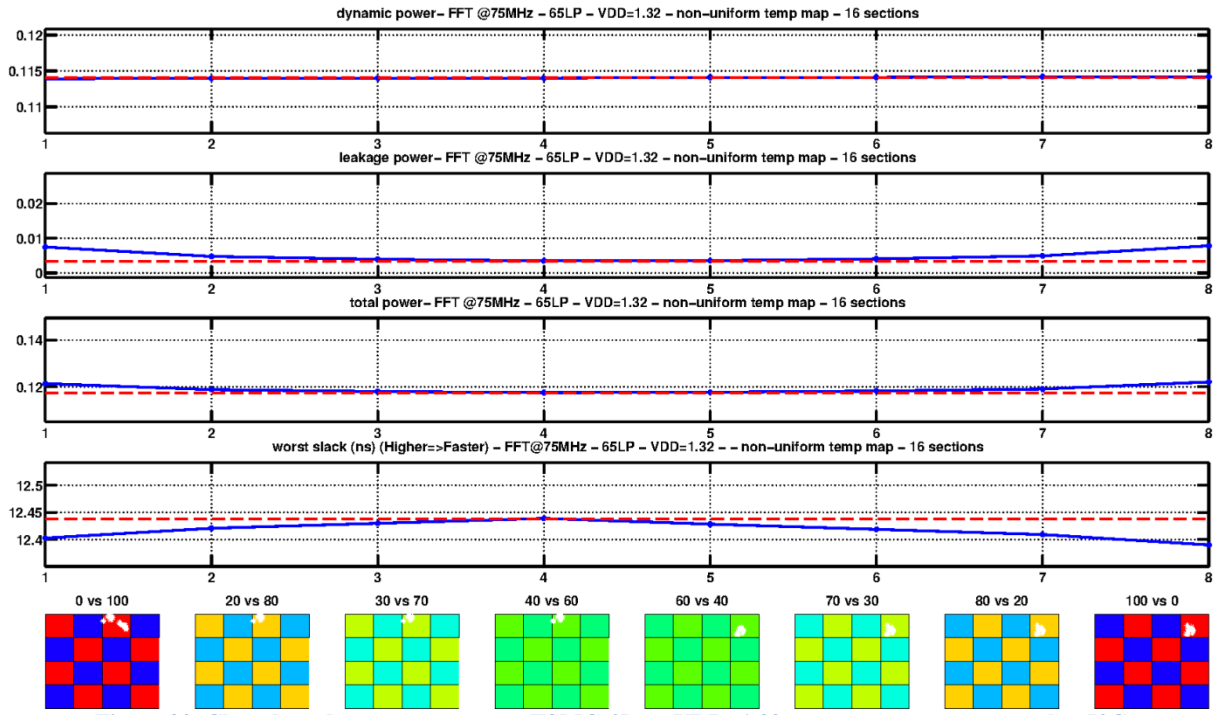


**Figure 20. Chess-board temperature map, TSMC 65nm, VDD=1.32v, average temperature value 50C.**

The plots for chess-board temperature pattern for TSMC 40nm technology is very similar to the presented above figures and so, we are not presenting them here.

Now we show the effect of having horizontal or vertical on-die gradients. We present these results of 40nm technology node.

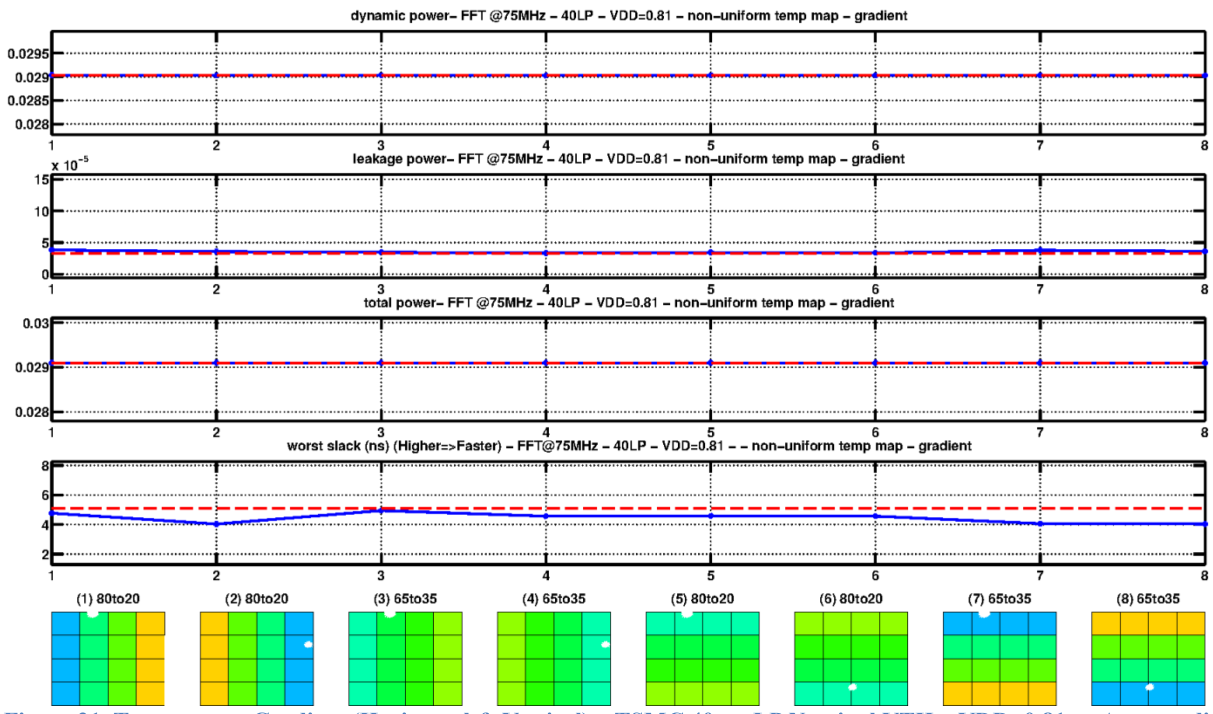


**Figure 21. Temperature Gradient (Horizontal & Vertical) – TSMC 40nm LP Nominal VTH – VDD=0.81v – Average die temperature 50 C.**

As we can see in Figure 21, the amount of power values for different test patterns is almost in line with the estimated power value at 50C, however for slack this is not true.

The locations of critical timing paths are usually near to colder areas of the chip. As we see by changing the temperature pattern the location of critical paths change significantly. For 65nm design we have the same concepts and so we no more represent it here.

Figure 22 is showing the same test as Figure 21, however we have increased the VDD to 1.21v. As we can see in Figure 22 , when VDD is at its maximum, critical paths happen usually at hotter areas of the chip.

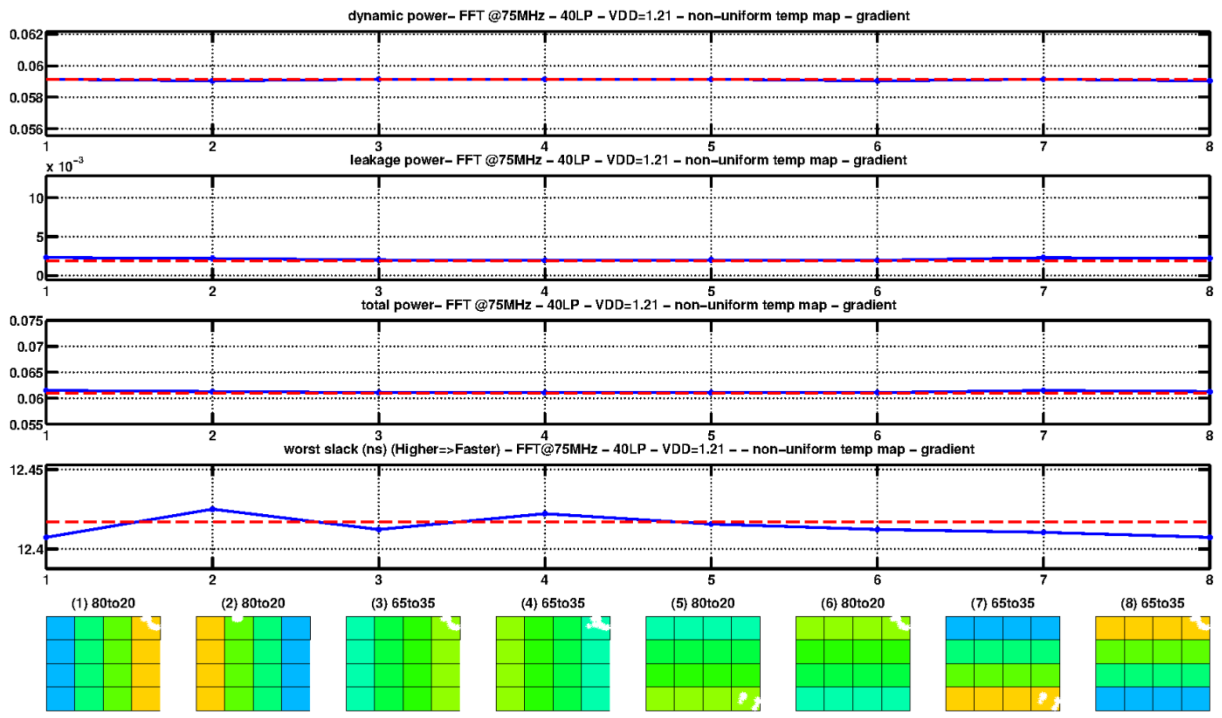


Figure 22. Temperature Gradient (Horizontal & Vertical) – TSMC 40nm LP Nominal VTH – VDD=1.21v – Average die temperature 50 C.

We now present examples of bell-shaped patterns for temperature and evaluate design parameters for it. Here we show the results for 65nm technology nodes.

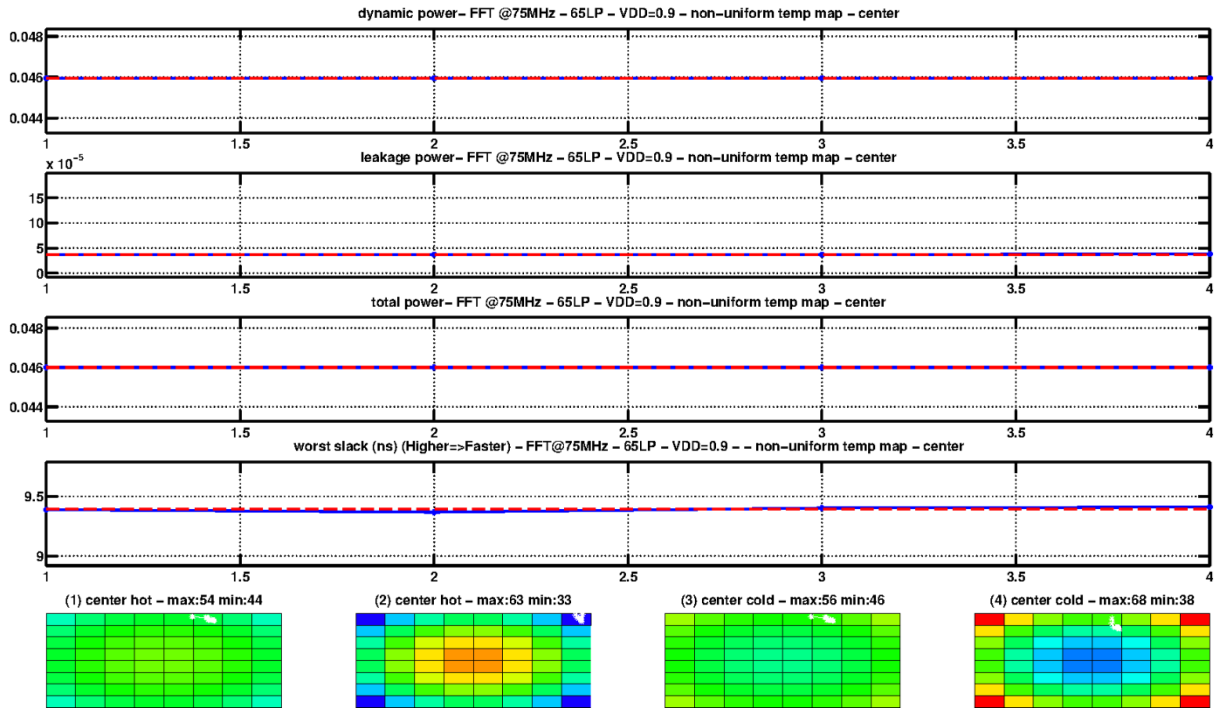


Figure 23. Bell-shaped temperature map, TSMC 65nm , VDD=0.9v

In Figure 23, the first two temperature maps represent a bell which has its maximum at the center. In the other two, the center is minimum. Two different temperature ranges are evaluated. As we can see for power values and slack, they are almost in line with the values at 50C. This hints us that for normal situations in which the chip has a usual bell-shaped temperature map, we may not need to be worried about the effect of temperature variation on accuracy of slack and power calculation.

As we can see in Figure 23, the physical location of timing critical path changes by temperature map. In fact, it moves nearer to colder locations of the chip.

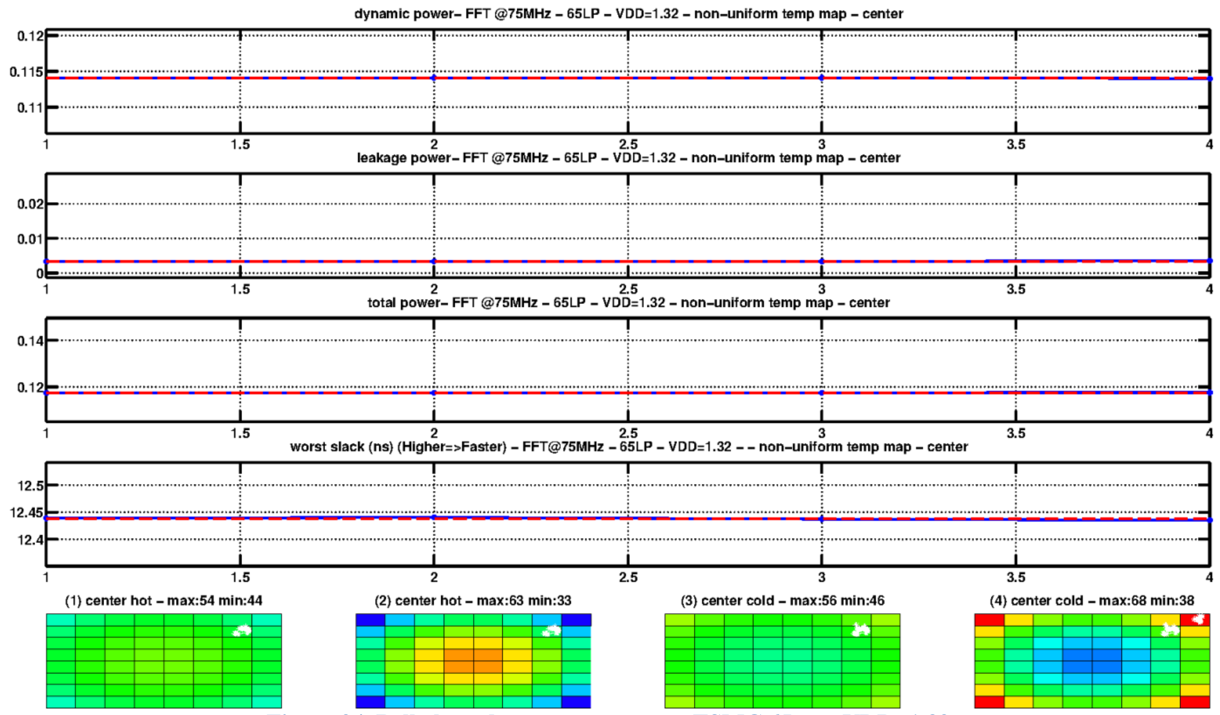


Figure 24. Bell-shaped temperature map, TSMC 65nm , VDD=1.32v

In Figure 24 the voltage is increased thus as we can see the timing critical path moves as near as possible to hotter places of the chip.

## 5. Self-Heating

We now represent results for a sample case in which we calculate design parameters considering design's self-heating effect.

In this example, the initial point for analysis is the ambient temperature of 25C. Our tool then calculates power/timing values of the design iteratively. In the first iteration it is supposed that the entire chip die has a unique value of 25C. Power of the design is calculated based on this assumption and thermal simulation is done. The temperature map obtained from thermal simulation will be done as the initial temperature map of the design in the next iteration.

The number of iterations is currently a user defined variable. In this test we perform 8 iterations. Figure 25 shows the result of this simulation.

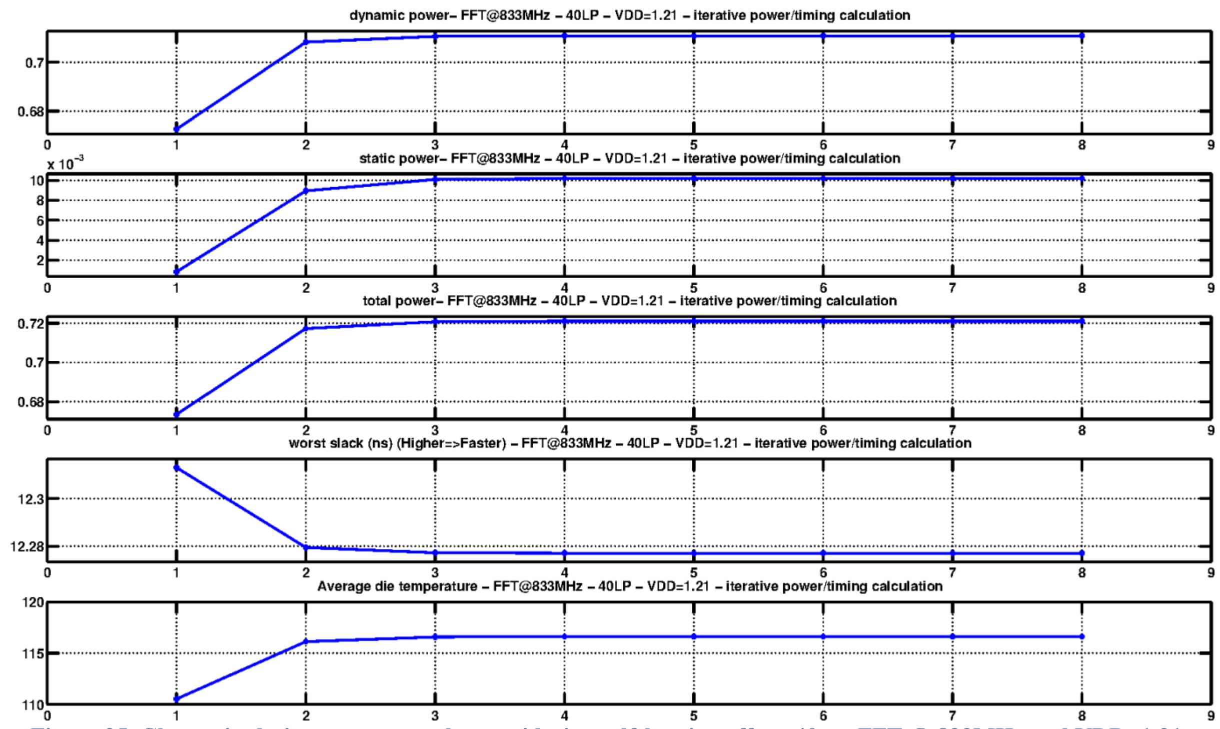


**Figure 25. Change in design parameters by considering self-heating effect, 40nm FFT @ 833MHz and VDD=1.21v.**

During the test shown in Figure 25, we have run FFT at 833MHz. This is because we wanted the module to consume a high power so that we can better see the self-heating effect. As we can see, the initial estimated slack for the circuit is different than the slack value considering self-heating effect. Moreover, the initial estimated averaged chip temperature is different than the real chip temperature by more than 5C degrees. In fact, as we can see in the figure, the convergence of power/timing parameters happens just after 3 iterations.

## 6. Conclusions and Remarks

We presented the effect of temperature variation on estimated power/timing parameters of a sample design. We studied the sample design at both of 65nm and 40nm technology nodes and provided comparisons between the speed of the circuit and its power consumption under various chip temperature and supply voltage conditions.

We used a standard tool flow based on Synopsys and developed our own tool-set which integrates into the flow seamlessly and provides us with the needed estimates. Our tool-set is compatible with standard design file formats and understands them. No conversion into any especial new format is required.

It should be noted that the process of power/timing calculation for a sample digital circuit in deep sub-micron designs, can be done in several different ways, which results in several different levels of accuracy. In this work we used a standard flow described in Synopsys documentation [7] for timing/power calculations. It is possible to increase the accuracy of results by taking various on-chip variation effects into consideration while doing timing/power analysis. For this purpose, suitable standard cell libraries should be also available to provide the tool with required data. This is however, more involved, and outside the scope of this research.

## 7. References

[1] P. A. Milder, "Spiral Project: DFT/FFT IP Core Generator," [Online]. Available: http://www.spiral.net/hardware/dftgen.html.

[2] Synopsys, "CCS Timing," *Synopsys Technical White Papers,* 2006.

[3] Synopsys, Synopsys Design Compiler User Guide, Version F-2011.09-SP2, December 2011.

[4] Synopsys, IC Compiler Implementation User Guide, Synopsys, Version F-2011.09-SP4, March 2012.

[5] "Impact of thermal effects on performance, power, and leakage trade-offs in advanced digital CMOS circuits, Therminator Deliverable D3.2.1," 20-Feb-2012 .

[6] G. Yip, "Expanding the Synopsys Prime-Time Solution with Power Analysis," June 2006., doc type?

[7] Synopsys, "PrimeTime® Fundamentals User Guide, Version D-2010.06," June 2010.

[8] W. Huang, "HotSpot: a compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* May, 2006. vol? pp?